UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Doble Grado en Ingeniería Informática y Matemáticas

TRABAJO FIN DE GRADO

Aplicaciones Móviles y Google Analytics para Investigaciones en el Campo de la Psicología

Ana Alonso Acosta
Tutor: Eduardo Boemo

MAYO 2018

Aplicaciones Móviles y Google Analytics para Investigaciones en el Campo de la Psicología

AUTOR: Ana Alonso Acosta
TUTOR: Eduardo Iván Boemo Scalvinoni

Dpto. TEC
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Mayo de 2018

Resumen

Este trabajo de fin de grado es una incursión en un proyecto del mundo real relacionado con la psicología. Hoy en día la informática es una herramienta de la que cada vez se hace más uso en otros campos para aprovechar el gran potencial del que dispone. En este caso la aplicación desarrollada es para el Departamento de Psicología de la Universidad Complutense de Madrid (UCM). El tema de investigación es la regulación emocional; en otras palabras, tener un conocimiento y un control sobre las propias emociones.

Para el desarrollo de esta aplicación se ha utilizado ampliamente la documentación oficial de Android. Se ha realizado una implementación ampliable, pensando en futuras versiones. La escalabilidad se refleja en el uso de elementos como una base de datos que podría ser prescindible para un modelo a escala pero no para una aplicación más grande en el mundo real. También se ha utilizado el patrón de diseño reconocido llamado Modelo-Vista-Controlador. Se ha puesto especial detalle en el control de errores para hacer que la experiencia de usuario sea lo más agradable posible. Un aspecto que también ha sido importante a la hora de desarrollar la aplicación ha sido la estética de esta ya que está comprobado que este aspecto afecta mucho a la valoración de un usuario y a su fidelidad hacia ella.

Palabras clave

Aplicaciones Móviles, Android, Bases de Datos, Modelo-Vista-Controlador, Control de Emociones, Psicología.

Abstract

This Bachelor Thesis consists of a first approach to a real world project related to psychology. Nowadays computer science is a tool which is being used more and more in different fields to make the most out of its potential. In this Thesis an app for the Psychology department of the Universidad Complutense de Madrid (UCM) was developed. The research field for which this app is going to be used is emotional regulation, in other worlds, having knowledge and control of one's own emotions.

The official Android documentation has been extensively used to develop this app. The implementation made takes into account the future development of the app. The scalability is implicit in the choice of using a database. This could be avoided for a smaller model, but not for a real world use. Also, the well known design pattern Model-View-Controller has been used. Special emphasis has been put in dealing with errors to make user experience as smooth as possible. An important detail that has been taken into account during the development of the app is aesthetics, because it has been proved that this greatly affects the assessment a user makes of an app and its fidelity to it.

Keywords

Mobile Apps, Android, Databases, Model-View-Controller, Emotion Control, Psychology.

Agradecimientos

A Eduardo, por darme la oportunidad de hacer un TFG en el que pudiera sentirme implicada.
A aquellas personas de mi vida que me aportan cada día algo valioso.
A Dani por darme fuerzas para seguir.
A mis padres, en especial a mi madre que me lo ha dado todo.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1 Introducción

1.1 Motivación

Los tratamientos para trastornos mentales, pueden ser intervenciones psicológicas, farmacológicas o programas de autoayuda. En España existe una importante escasez de recursos humanos en la sanidad pública (en promedio, se hace una sesión de una hora mensual por paciente). Esto dificulta la continuidad en los tratamientos y el seguimiento de los pacientes.

Como alternativa a lo anterior, se están empezando a usar *apps* para monitorizar o registrar el estado del paciente. Es decir, no reemplaza al terapeuta, pero sí ayuda a mantener el contacto con el mismo.

Las principales aplicaciones de las tecnologías de la información y las comunicaciones a la atención psiquiátrica son:

- Llamadas o SMS.
- Videollamadas.
- Intervenciones a través de la Web.
- Integración de sensores para la monitorización de medidas de los pacientes.
- Monitorización a través de las redes sociales.
- Juegos.

La universalización de los *smartphones* ha abierto una nueva línea de acción: *apps* para el uso en el campo de la salud mental.

1.2 Objetivos

El objetivo de este TFG es realizar un primer prototipo de una *app* para la regulación emocional. El mismo ha sido especificado y será evaluado dentro de un programa de investigación del Departamento de Psicología de la Universidad Complutense de Madrid (UCM).

Desde el punto de vista técnico, la *app* debe tener las siguientes funciones:

- Menú principal con acceso a otras tres pantallas de la aplicación.

- Cuestionario inicial que se repetirá cada X usos de la aplicación en el que aparecen preguntas que el usuario debe puntual de 1 a 5.
- Pantalla de etiquetación en la que el usuario podrá guardar una selección de emociones para describir su estado de ánimo al que podrá agregar una breve descripción.
- Pantalla de entrenamiento en etiquetación emocional. En ella deberá mostrarse la descripción de una situación y una lista de emociones. Tras la selección de una emoción por parte del usuario, este recibirá información sobre si la elección fue correcta.
- Pantalla de historial en la que el usuario podrá consultar todas las etiquetaciones guardadas hasta el momento.

2 Estado del Arte

El *Institute for Healthcare Informatics* indica que existen cerca de 300K aplicaciones móviles relacionadas con la salud en las principales plataformas de descarga y cerca del 40%, son *apps* para la monitorización de pacientes [1].

Las dos categorías principales en las que se clasifican las *apps* de salud son:

- Salud-Bienestar: Está dirigida a cualquier usuario; no sólo aquellos que pueden necesitar tratamiento. No pretende conseguir ninguna curación. En muchos casos carecen de bases técnicas y pueden tratar temas de autoayuda, algunos muy cercanos a las pseudociencias. En este TFG se han dejado de lado el estudio de este tipo de apps.

- Medicina: Ofrecen funcionalidades avaladas científicamente. Están orientadas a pacientes o estudios médicos. No están disponibles en la *Play Store* o *iTunes*. Su uso se realiza bajo control médico. Este tipo de *apps* son certificadas en EEUU como dispositivo médico [2].

La *app* desarrollada en este TFG se pretende ubicar dentro de la segunda categoría. Será probada por un equipo del Departamento de Psicología de la Universidad Complutense de Madrid (UCM), liderado por el profesor G. Hervás. Los detalles relacionados con el campo de la Psicología escapan al ámbito de este TFG, que se centra en los aspectos técnicos y de programación. Los lectores interesados en los detalles de investigación relacionados con esta *app* pueden consultar la siguiente referencia [3].

3 Diseño

El diseño de esta aplicación está basado en el patrón Modelo-Vista-Controlador (MVC), un patrón ampliamente conocido en programación. En este patrón el componente del *modelo* es el que alberga los datos que se utilizan en la aplicación, los cuales se pueden tanto consultar como modificar. La *vista* como su nombre indica es el componente que ve el usuario y utiliza los datos del modelo para mostrarlos de una manera determinada. Por último, el componente del *controlador* es el que media entre los dos anteriores, ante cambios en la *vista* o ante acciones del usuario, actúa ya sea sirviendo nuevos datos o modificando aquellos que se encuentran en el *modelo*.

Este modelo es uno de los primeros que se diseñaron cuando aparecieron las interfaces de usuario como componente en sí mismo, de ahí que haya evolucionado en distintas variantes y hoy en día no se utilice en su forma más pura. En particular, en las aplicaciones móviles la vista y el controlador tienen una estrecha relación por lo que terminan estando menos diferenciadas que en la definición del patrón MCV. En la siguiente figura se puede ver una clasificación aproximada de las clases del proyecto y del papel que juegan en el patrón.

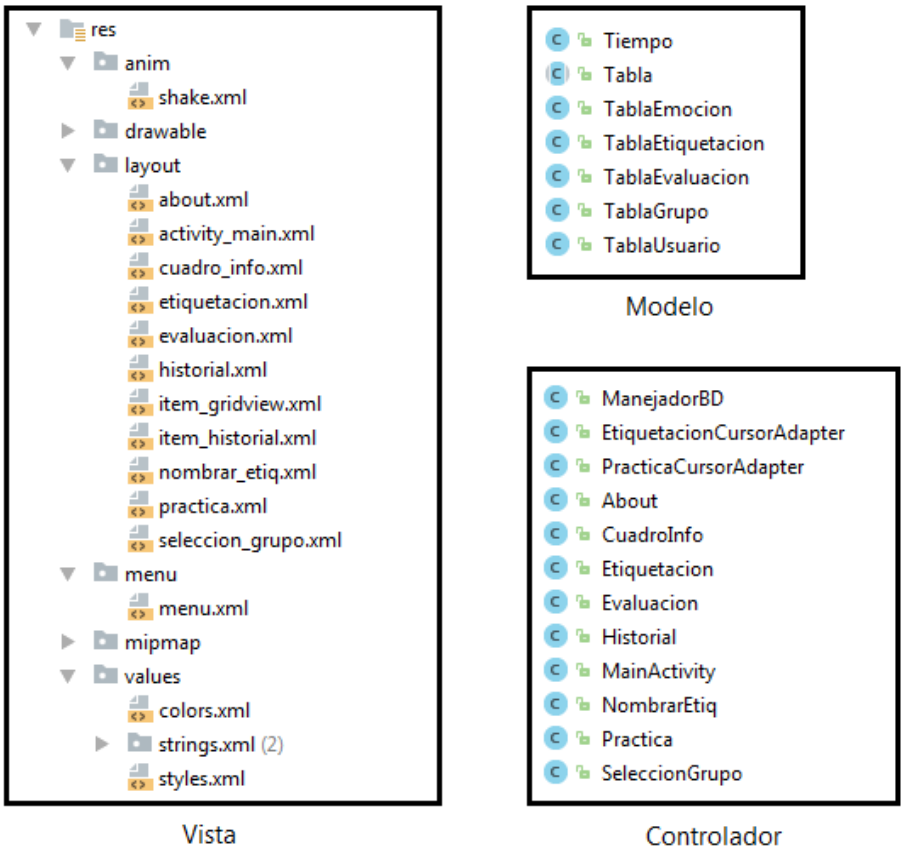

Figura 3-1: Diseño del proyecto

Como se ha mencionado antes, controlador y vista están fuertemente relacionados por lo que la separación no está tan definida como en el anterior esquema. En este caso las clases del controlador tienen funcionalidad que pertenece al componente de la vista.

3.1 Modelo

Las clases que pertenecen al modelo son aquellas que están directamente relacionadas con el tratamiento de datos. En el caso de esta aplicación, principalmente son aquellas que modelan el esquema de la base de datos.

3.1.1 Clases que representan tablas

Existe una clase abstracta "Tabla" que tiene los atributos que serán comunes para todas las demás clases que representen una tabla de la base de datos. El resto de clases que representan una tabla, tienen definida la estructura de la tabla correspondiente para realizar operaciones sobre ella.

3.1.2 Clase de formato de datos

Por último en este componente también encontramos la clase "Tiempo" que no modela una tabla de la base de datos, pero se encarga de darle un formato a la fecha antes de guardarla en la base de datos.

3.1.3 Base de datos

En el siguiente esquema se muestra la base de datos de la aplicación. Aunque esta no sea un fichero que se cree en Android Studio (el entorno de programación), como los .xml o los .java, es parte del modelo de la aplicación y se crea en el dispositivo a partir de código Java.
Este esquema se ha obtenido de una herramienta online [4] que permite obtener un diagrama UML de una base de datos, a la vez que obtener el código SQL para generar la base de datos diseñada para diferentes motores de bases de datos.

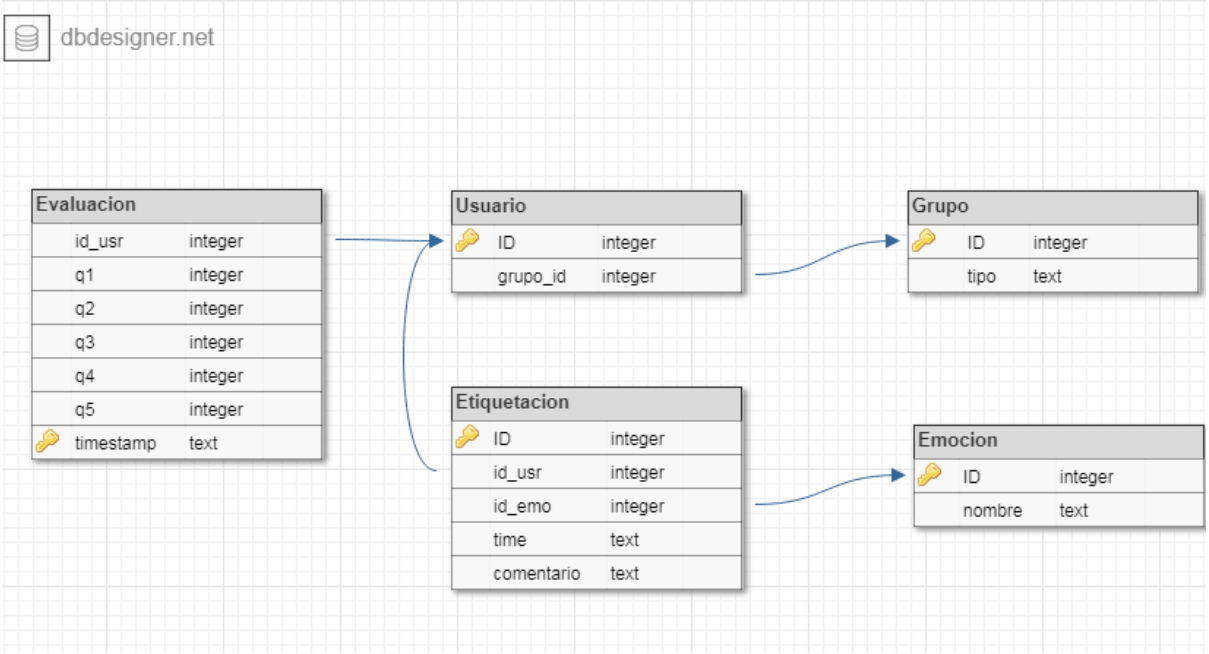

Figura 3-2: Esquema de la base de datos

Como se puede comprobar es una base de datos pequeña que consta de cinco tablas. Actualmente una base de datos puede ser una herramienta más potente de lo que necesita la aplicación para funcionar pero esta implementada con miras al futuro. Una base de datos es la mejor estructura para guardar grandes cantidades de información y la manera más homogénea para almacenar información de distintas fuentes (usuarios). También es una estructura con mucho potencial para elaborar estadísticas a partir de los datos contenidos en ella.

3.2 Vista

Es el componente que cuenta con más ficheros pero estos son de menor complejidad. En ellos se encuentra definida la posición, tamaño y demás aspectos visuales de las distintas pantallas que forman parte de la aplicación. Todos los elementos que pertenecen a este componente son ficheros XML y se encuentran en la carpeta "res" (de *resources*). La clasificación que hace Android de este tipo de elementos es la siguiente.

3.2.1 Anim

Esta carpeta contiene los archivos que describen animaciones para elementos de la aplicación. En este caso la única animación del proyecto es una traslación para mostrar un mensaje de error.

3.2.2 Drawable

Drawable alberga todas las imágenes utilizadas en la aplicación además de elementos visuales simples, como formas con colores que se utilizan muchas veces como fondo de otros elementos.

3.2.3 Layout

En el directorio con este nombre podemos encontrar principalmente la definición de la vista de las pantallas que componen la aplicación. En general, estos ficheros definen estructuras con distintos componentes. En el caso de esta aplicación existe un *layout* por pantalla además de dos extras que son elementos que forman parte de alguno de los anteriores. Estos *layouts* que no representan una pantalla de la aplicación, proporcionan la estructura para los elementos de una *gridview* o una *listview*, que son componentes que sirven para crear listas dinámicas sobre las que uno se puede desplazar.

3.2.4 Menu

En este directorio se guardan los menús que pueda tener la aplicación, es decir la barra superior en la que pueden aparecer distintas opciones. En el caso de esta aplicación se guarda el menú que se muestra en la pantalla principal que contiene el botón para ir a la pantalla “about”.

3.2.5 Mipmap

En esta carpeta se guardan los iconos (para diferentes densidades de pantalla) con los que aparecerá representada la aplicación en los dispositivos móviles.

3.2.6 Values

En este directorio encontramos los archivos styles.xml, colors.xml y strings.xml. En el primero se define el uso de colores por defecto (como el color de fondo) para la aplicación para que así no sea necesario elegirlos en todas las pantallas sino que solo haga falta especificar casos excepcionales. En el fichero colors.xml se asocian nombres a colores en hexadecimal para que el uso de estos sea más transparente. Por último, en strings.xml se encuentran todas las cadenas de caracteres que se van a utilizar en la aplicación. Tenerlas en un archivo propio en vez de escritas dentro del código permite hacer que la aplicación soporte distintos idiomas.

Cabe destacar que estas no son las únicas carpetas que puede haber en el apartado "res" para un proyecto, pero son las únicas que se han necesitado para este.

3.3 Controlador

El componente del controlador es aquel que tiene mayor peso dentro de esta aplicación ya que es donde recae mayor funcionalidad porque comunica la vista con el modelo. Además en este caso es el componente en el que se encuentra cierto solapamiento con la vista y el modelo.

3.3.1 Clases que interactúan solo con la vista

Las funcionalidades directamente relacionadas con la vista que se encuentran en estas clases son sobre todo aplicar algunos estilos a los textos que aparecerán en pantalla. En cuanto a las funciones que propiamente son del controlador encontramos las funciones *listener* que esperan a que ocurra un determinado evento (como pulsar un elemento o mantenerlo pulsado) para ejecutar el código en su interior (como pasar a una nueva pantalla de la aplicación o cambiar parcialmente la vista de la actual).

Bajo la descripción anterior se encuentran dos clases muy parecidas "PracticaCursorAdapter" y "EtiquetacionCursorAdapter". Ambas son utilizadas a través de otras dos que también pertenecen al controlador ("Practica" y "Etiquetacion"). Son las encargadas de enlazar datos extraídos de la base de datos que les proporciona otra clase ("Practica" y "Etiquetacion" respectivamente) con su vista. Esta forma parte de un elemento de tipo *gridview* dentro de un *layout*. El resto de clases que pertenecen a esta clasificación ("About", "CuadroInfo", "MainActivity" y "NombrarEtiq") modifican elementos más sencillos de sus respectivos *layouts*.

3.3.2 Clases que interactúan solo con el modelo

En el proyecto solo hay una clase que cumpla con esta descripción y es "ManejadorBD". Es la única clase del controlador que no tiene ninguna relación con la vista. Su función es hacer de intermediario con la base de datos, es decir, a través de ella se pueden hacer consultas o modificaciones.

3.3.3 Clases que interactúan con la vista y el modelo

En este apartado están las clases que manipulan elementos que se muestran en las pantallas y que además realizan consultas a la base de datos para mostrar esa información (como en "Historial", "Practica" y "Etiquetacion") o guardan nueva información en la base de datos (como en "Etiquetacion", "Evaluacion" o "SeleccionGrupo").

4 Desarrollo

El proceso que se siguió para desarrollar la aplicación se basó en tener, siempre que fuera posible, una versión ejecutable de esta. Lo primero siempre era definir la vista a la hora de crear una nueva pantalla de la aplicación. Después de eso, se implementaba la funcionalidad asociada a dicha pantalla, total o parcialmente, dependiendo de la complejidad y el número de funcionalidades independientes, y por último se probaba en un dispositivo móvil si el funcionamiento era correcto.

4.1 Procesos principales

Para ilustrar mejor los procesos más importantes de esta aplicación se muestra a continuación un mapa de navegación entre pantallas.

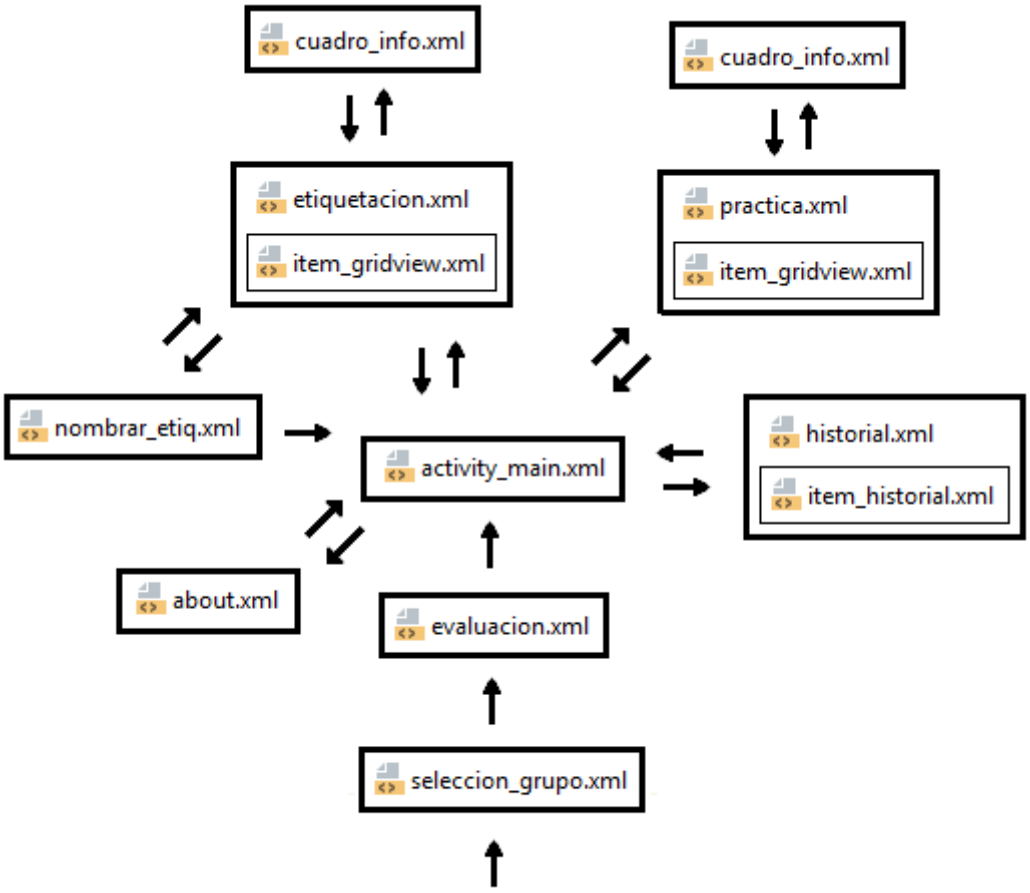

Figura 4-1: Mapa de navegación

Los procesos con mayor complejidad de la aplicación se encuentran en el ciclo activity_main.xml → etiquetacion.xml → nombrar_etiq.xml → activity_main.xml. Si se pasa por esta secuencia de pantallas es porque desde la pantalla principal (activity_main.xml) se ha pulsado la opción de "Etiquetación" y en esta pantalla (etiquetacion.xml) se han seleccionado un conjunto de emociones (al menos una) y se ha pulsado el botón de guardar. Tras esto aparece un cuadro (nombrar_etiq.xml) para poner una breve descripción a la etiquetación que se va a guardar y tras pulsar en la opción de guardar se vuelve a la pantalla principal.

Este proceso es posiblemente el más complicado de la aplicación porque requiere del paso de información entre actividades/clases (cada uno de estos *layouts* tiene una clase asociada que maneja la información) además de ciertos controles de errores. Más concretamente desde la clase correspondiente a nombrar_etiq.xml se guarda la descripción introducida (si se ha escrito algo) en un campo de la clase asociada a etiquetacion.xml. Además se ha implementado un método para que al finalizar la actividad en la que se introduce la descripción, eligiendo la opción de guardar, la actividad en la que se habían seleccionado el conjunto de emociones guarda estas junto a la recién proporcionada descripción, en la base de datos.

Otro proceso principal, pero de menor grado de complejidad es la gestión de las soluciones en la pantalla practica.xml. En esta pantalla se describe una situación hipotética y se proporciona una lista de emociones (elementos de una *gridview*). También existe una lista predefinida de emociones que conforman la solución para cada situación. Teniendo esta información en cuenta, el funcionamiento de la pantalla es el siguiente. Al seleccionar una emoción que pertenece a la solución de la actual situación, aparece su descripción en un cuadro en la parte inferior de la pantalla y el elemento de la *gridview* con el nombre de la emoción que se ha seleccionado se pone de color verde. Si por el contrario la emoción seleccionada no pertenece a la solución, solamente aparecerá la descripción en el cuadro inferior. A parte de esta dinámica, pero en relación con ella, existe un botón en esta pantalla para revelar la solución de la situación que está actualmente en pantalla. Tras pulsar este botón se muestran en verde todas las emociones que eran parte de la solución (como si las hubiéramos seleccionado). Debido a que estos dos procesos interactúan con los mismos elementos se necesita hacer un seguimiento de las emociones que ya han sido seleccionadas y hacer esta información accesible desde las dos partes.

Como se ha comentado antes, la compartición de información entre actividades puede llegar a ser compleja. Este caso no es el mejor ejemplo de ello, porque las dos clases involucradas no eran independientes, sino que una de ellas creaba la otra, por lo que podía acceder a los campos de la segunda que estuviesen declarados como públicos (concretamente la clase asociada a practica.xml creaba la que gestionaba la lista de emociones).

En orden descendente de complejidad otro proceso importante sería la visualización del historial. En este caso ya no hay dos clases que tengan que compartir información

bidireccionalmente. Ahora existen dos clases implicadas en este proceso, la que se encarga de casi todos los elementos de la pantalla y la que gestiona la lista de etiquetaciones extraídas de la base de datos. Pero la diferencia con el caso anterior es que la clase creadora proporciona los datos de la base de datos a la clase que va a rellenar la lista (*listview*) y no necesitan volver a comunicarse más.

Por último, el funcionamiento de la clase asociada a evaluación.xml es el siguiente. Esta es una pantalla de baja complejidad ya que consta de un test de cinco preguntas a las que hay que contestar con un número del 1 al 5 y solo es necesario hacer un control para asegurarse de que todas las preguntas han sido contestadas.

4.1.1 Ciclo de vida de la base de datos

La interacción de la aplicación con la base de datos descrita en el apartado 3.1.3 es la siguiente.

La tabla "Grupo" tiene tantas entradas como grupos distintos para clasificar usuarios se quieran tener. Estos grupos hay que definirlos en el código de la aplicación y a partir de este se genera la tabla. Esta tabla se crea la primera vez que se utiliza la aplicación y no se vuelve a modificar.

La tabla "Usuario" está relacionada con la tabla "Grupo" porque tiene una referencia al grupo que eligió el usuario la primera vez que ejecutó la aplicación. Esta tabla solo tiene una entrada que se crea tras la elección de grupo y no se vuelve a modificar.

La tabla "Emocion" se crea a partir de información que hay en el código de la aplicación y es la que guardará el listado de las emociones disponibles en la aplicación.

Las dos tablas que faltan por mencionar, la tabla "Evaluacion" y la tabla "Etiquetación", se parecen entre sí porque son las dos tablas que sí se modifican guardando información sobre el usuario a lo largo del uso de la aplicación.
En "Evaluacion" se guardan las respuestas al test que aparece cada X usos de la aplicación y en "Etiquetacion" se guardan la hora, la fecha, la descripción proporcionada y el conjunto de emociones que seleccionó el usuario para guardar en un momento dado.

4.2 Dificultades y soluciones

En esta sección se comentan algunas de las complicaciones surgidas a lo largo del proceso de desarrollo de la aplicación y cómo finalmente se solucionaron.

Una funcionalidad interesante de la pantalla evaluacion.xml es que aparece cada X usos de la aplicación. El número de usos necesarios para que vuelva a lanzarse la actividad

asociada a esta pantalla se puede cambiar en el código. Este comportamiento se ha conseguido a través de las preferencias de la aplicación, que son valores que se pueden guardar y se mantienen aunque se cierre la aplicación [5].

El elemento que fue más complicado de diseñar e implementar de la aplicación, fue la clase que enlazaba los datos con la *gridview*. Esta vista de cuadrícula aparece en dos lugares de la aplicación, en practica.xml y etiquetación.xml, en ambos muestra la lista de emociones de la aplicación en forma de cuadrícula en la que se puede hacer *scroll*. Tanto la *gridview* como el elemento que sirve para enlazarla con los datos tienen una gran ventaja y es que al ser de alto nivel proporcionan una funcionalidad compleja con relativas pocas líneas de código. Su desventaja, por otro lado, reside también en que sean de alto nivel. Más concretamente el elemento que enlaza la *gridview* con los datos, utiliza funciones de las que no es sencillo conocer su funcionamiento solo leyendo la descripción de la documentación [6].

Entrando más en detalle, existen unos elementos que son de tipo *adapter*, que se utilizan para servir el contenido a este tipo de vistas dinámicas (*listview, gridview, recyclerview…*) siempre que este esté en forma de colección homogénea de datos. En el caso de la *gridview* de la aplicación, utiliza un *adapter* de tipo *cursor* que es el tipo de estructura que tienen los resultados devueltos por una consulta a una base de datos.

Para crear este *adapter* hace falta extender la clase *CursorAdapter* e implementar tres funciones como mínimo: el constructor, *newView* y *bindView*. En la definición de *newView* se dice que crea una nueva vista para contener el dato al que apunta el cursor y en la de *bindView* se puede leer que asocia una vista existente al dato al que apunta el cursor.

La primera idea que se barajó sobre el posible funcionamiento de estos métodos fue la siguiente. Cuando se accedía a la pantalla se creaban las vistas para todos los elementos que iban a ser mostrados (una vista por elemento) y se les asociaban a cada una el elemento correspondiente. Es decir, bajo esta hipótesis, estas dos funciones solo se ejecutaban en el momento de crear la pantalla tantas veces como elementos fuera a haber en la lista. Tras una serie de pruebas, utilizando el *log* del sistema [7] para poder hacer un seguimiento de las ejecuciones de estas dos funciones, se comprobó que no era así.

A través de los datos obtenidos en el *log* se comprendió finalmente la dinámica que seguía esta clase para la invocación de los métodos. El método *newView* se ejecutaba tantas veces como elementos fuera a haber simultáneamente visibles y el método *bindView* se ejecutaba cada vez que un elemento que no estuviera en pantalla fuera a ser mostrado. Es decir que con las vistas justas para rellenar la pantalla se iban cargando los elementos bajo demanda, según se hiciera *scroll* hacia arriba o hacia abajo. Tras comprender cuál era el funcionamiento real del componente su programación fue mucho más sencilla.

Otro elemento sobre el que hizo falta documentarse ampliamente fue la base de datos, más concretamente el manejador de la base de datos que es el elemento a través del cual se

canalizan las sentencias SQL. Al principio existían dudas sobre si el manejador de base de datos era el elemento que se necesitaba para interactuar con la base de datos y se llegó a investigar otro llamado proveedor de contenido. Al hacerlo se comprobó que este componente no era necesario en la aplicación que estaba siendo implementada ya que esta iba a tener toda la información en local. Un proveedor de contenido es un elemento que comunica la aplicación con servicios exteriores, como puede ser una base de datos si esta se encuentra en un servidor.

Tras descartar el proveedor de contenido como una posibilidad para la aplicación en desarrollo, quedó claro que un manejador de base de datos iba a colmar las necesidades de ésta. Este componente viene detallado en la documentación oficial [8]. La solución a la comunicación con la base de datos era en realidad sencilla, ya que bastaba con hacer una clase que extendiera SQLiteOpenHelper, una API muy potente para comunicarse con una base de datos en SQLite. El problema que existía en el momento de implementar dicha clase era que al ser de tan alto nivel parecía demasiado sencillo, ya que ni siquiera había que abrir una conexión con la base de datos, si no simplemente ejecutar las consultas.

4.3 Datos estadísticos

Para tener una idea del tamaño de todos aquellos elementos que han aparecido en esta memoria se han obtenido algunas estadísticas que lo ilustran.

Source File	Total Lines	Source Code Lines	Blank Lines
about.xml	98	87	11
activity_main.xml	117	109	8
AndroidManifest.xml	55	53	2
colors.xml	17	17	0
cuadro_info.xml	28	28	0
etiquetacion.xml	99	92	7
evaluacion.xml	402	346	56
fondo_cuadro_info.xml	19	16	3
historial.xml	66	59	7
ic_launcher_background.xml	170	170	0
ic_launcher_foreground.xml	34	34	0
item_gridview.xml	19	17	2
item_historial.xml	13	13	0
menu.xml	9	8	1
nombrar_etiq.xml	60	54	6
practica.xml	122	113	9
rounded_corners_item_sel.xml	13	13	0
rounded_corners_item.xml	19	16	3
rounded_corners_no_sol.xml	19	16	3
rounded_corners_sol.xml	19	16	3
rounded_corners.xml	19	16	3
seleccion_grupo.xml	77	72	5
shake.xml	11	10	1
strings.xml	149	149	0
strings.xml	149	149	0
styles.xml	12	10	2
toast_blanco.xml	19	16	3
Total:	**1834**	**1699**	**135**

Tabla 1: Líneas de código xml

En estos ficheros el 93% de líneas, son líneas de código y el restante 7% son líneas en blanco para hacer más fácil la lectura del código. En el caso de los ficheros xml no se han añadido comentarios ya que al describir elementos de la vista son bastante autoexplicativos. Además como es un lenguaje más descriptivo, una rápida búsqueda del componente en la documentación oficial es suficiente para aclarar las dudas.

Source File	Total Lines	Source Code Lines	Comment Lines	Blank Lines
About.java	77	51	16	10
CuadroInfo.java	75	44	23	8
Etiquetacion.java	217	147	48	22
EtiquetacionCursorAdapter.java	85	41	35	9
Evaluacion.java	160	107	39	14
ExampleInstrumentedTest.java	26	15	6	5
ExampleUnitTest.java	17	9	5	3
Historial.java	190	115	53	22
MainActivity.java	210	132	51	27
ManejadorBD.java	74	47	20	7
NombrarEtiq.java	103	66	26	11
Practica.java	255	161	70	24
PracticaCursorAdapter.java	91	44	38	9
SeleccionGrupo.java	110	75	21	14
Tabla.java	15	7	7	1
TablaEmocion.java	23	11	8	4
TablaEtiquetacion.java	29	16	9	4
TablaEvaluacion.java	34	19	12	3
TablaGrupo.java	20	11	6	3
TablaUsuario.java	24	12	7	5
Tiempo.java	35	17	14	4
Total:	**1870**	**1147**	**514**	**209**

Tabla 2: Líneas de código java

En cuanto a los datos sobre código Java el 61% de líneas forman parte del código como tal, el 11% son líneas en blanco para ayudar a la legibilidad del código y por último el 28% son líneas de comentarios. Se puede destacar el alto contenido de comentarios. Uno de los propósitos a la hora de crear esta aplicación fue la elaboración de suficientes comentarios para que su continuidad por parte de cualquier otra persona que tuviera los conocimientos teóricos suficientes, fuera lo más sencilla posible.

Después de haber comentado los datos de ambas tablas por separado, merece la pena cruzar los datos y analizar si existe alguna relación. Se puede sacar del análisis que la complejidad visual y la funcional no tienen por qué ir de la mano ya que la pantalla (vista) que más líneas de código requirió fue evaluacion.xml y sin embargo la funcionalidad (controlador) que fue más costosa de desarrollar fue Practica.java. Pero aunque la relación entre líneas de código empleadas para la vista y el controlador no sea directa, sí existe. Si nos fijamos en pantallas y funcionalidades que sean importantes en la aplicación, por ejemplo, el historial, la pantalla principal y la de etiquetación, todas superan las cien líneas de código Java y las dos primeras también lo hacen en su archivo XML. Por tanto, aunque la relación no se dé en todas las parejas vista-controlador tampoco es totalmente inexistente.

Estas tablas con datos estadísticos sobre el código de una aplicación Android han sido obtenidas con un *plugin* que puede instalarse en Android Studio [9]. Este permite seleccionar las carpetas del proyecto sobre las que se hace el análisis de la composición del código y del tamaño en bytes de los distintos componentes.

El tiempo empleado en la elaboración de este TFG se encuentra en torno a las 350 horas de las cuales al principio el mayor porcentaje se dedicó a leer para aprender sobre las particularidades de Android respecto a Java y más brevemente sobre XML. Después de haber adquirido una visión general, se empezó a dedicar más tiempo a la programación hasta que aparecía un elemento nuevo sobre el que se realizaba la pertinente investigación antes de plasmarlo en código.

5 Integración, pruebas y resultados

Las pruebas que se han llevado a cabo para esta aplicación tenían el objetivo de comprobar su correcto funcionamiento. La prueba dentro de un estudio de psicología, escapa al ámbito de este TFG.

Se ha utilizado el *log* del sistema para comprobar datos guardados y extraídos de la base de datos y también como método de depuración para imprimir variables cuando el funcionamiento no era el adecuado. También se ha probado la aplicación en cierta variedad de dispositivos para comprobar la adaptación de los elementos visuales a diferentes tamaños de pantalla. Se han hecho pruebas en dispositivos de 3,8'', 5,5'', 6'' y 8''. Hasta tamaños de 6 pulgadas la distribución de los elementos es agradable a la vista pero para el dispositivo de 8 pulgadas empiezan a notarse algunas pantallas algo vacías. La mayoría de los móviles están por debajo de las 8 pulgadas así que no supone un problema, de hecho el dispositivo de 8 pulgadas que utilizamos fue una *tablet*. Por último, cuando la aplicación fue operativa se ejecutaron todos los caminos que podía tomar un usuario para poner a prueba toda la funcionalidad.

6 Conclusiones y trabajo futuro

6.1 Conclusiones

Al finalizar este TFG se ha obtenido una versión funcional de la aplicación que se ha descrito y analizado a lo largo de esta memoria. Por lo que se puede decir que ha sido una aproximación a un proyecto de la vida real. A través de este proyecto he adquirido los conocimientos que tengo de Android y he tenido la experiencia de tratar con el cliente para el que finalmente iba a ser la aplicación, haciendo una toma de requisitos y una negociación en el caso de que no pudieran satisfacerse dichos requisitos. Se ha puesto especial cuidado en implementar todos los componentes con las herramientas que permitieran hacer las menores modificaciones en el código existente para poder desarrollar funcionalidades nuevas.

6.2 Trabajo futuro

El uso de Google Analytics, inicialmente planificado para esta primera versión, se ha pospuesto y será integrado en una segunda versión de la *app*, que se desarrollará después de la realización de las pruebas de campo. Esta herramienta de Google proporciona datos sobre el uso de la aplicación por parte del usuario como con qué frecuencia la utiliza, en qué pantallas pasan más tiempo o desde qué tipo de dispositivo se está ejecutando.

Otro trabajo futuro está relacionado con la base de datos. Se implementará un servidor central que almacene la información de todos los usuarios y que será de gran ayuda a la hora de elaborar resultados. El hecho de que se implementara una base de datos para este proyecto está relacionado con esta vía de desarrollo.

Otra mejora será la posibilidad de soportar varios idiomas en la aplicación. En cuanto a este aspecto, también se ha tenido en cuenta y se ha elaborado un archivo strings-es.xml que da pie a continuar añadiendo versiones en diferentes idiomas de los textos de la aplicación.

Finalmente, se implementará en la siguiente versión un sistema de notificaciones que avise al usuario cuando lleve un tiempo determinado sin utilizar la aplicación.

Referencias

[1] "The Growing Value of Digital Health", https://www.iqvia.com/institute/reports/the-growing-value-of-digital-health

[2] FDA Federal Drug Administration. Mobile Medical Applications – Guidance for Industry and Food and Drug Administration Staff. (2013), http://www.fda.gov/downloads/MedicalDevices/./UCM263366.pdf

[3] Hervás, G. (2011). "Psicopatología de la regulación emocional: el papel de los déficit emocionales en los trastornos clínicos". Psicología Conductual, 19(2), 347-372, http://0-search.proquest.com.cisne.sim.ucm.es/docview/993158755?accountid=14514

[4] "Online database schema designer", https://www.dbdesigner.net/

[5] "Save key-value data", https://developer.android.com/training/data-storage/shared-preferences#java

[6] "CursorAdapter", https://developer.android.com/reference/android/widget/CursorAdapter

[7] "Log", https://developer.android.com/reference/android/util/Log

[8] "Save data using SQLite", https://developer.android.com/training/data-storage/sqlite?hl=es-419

[9] "Statistic", https://plugins.jetbrains.com/plugin/4509-statistic

Glosario

API	Application Programming Interface
MVC	Modelo Vista Controlador
SQL	Structured Query Language
TFG	Trabajo de Fin de Grado
UCM	Universidad Complutense de Madrid
UML	Unified Modeling Language
XML	Extensible Markup Language

Anexos

A Manual de uso

Lo primero que se puede ver al ejecutar la aplicación por primera vez, es una pantalla que le pide al usuario que elija su grupo. Estos aparecen listados como grupo 1, grupo 2…Así con tantos grupos como haya. El nombre real que describe la naturaleza del grupo no se le muestra al usuario para no darle información que pudiera influenciarle. Esta pantalla aparece hasta que se elija un grupo.

Tras la elección de grupo se muestra un cuestionario con cinco preguntas sobre las actitudes del usuario/paciente cuando tiene sentimientos negativos. Este test no permite continuar hasta que se hayan contestado todas las preguntas y aparece cada X usos de la aplicación.

Finalizada la evaluación (el test) el usuario es redirigido a la pantalla principal de la aplicación, la que normalmente aparece al abrirla. En ella se pueden encontrar cuatro elementos.

El primero de ellos es el menú, donde aparece el título de la aplicación a la izquierda y un símbolo de información a la derecha. Al pulsar el símbolo de información aparece una pantalla con datos relativos a las partes involucradas en el diseño y desarrollo de esta aplicación.

El segundo elemento es el enlace a la pantalla de etiquetación. En esta pantalla, la primera vez que se abre aparece un cuadro informativo que después de aceptarlo, se puede volver a leer si se selecciona el símbolo de información que aparece al lado del título, arriba a la derecha. Debajo del título aparece la lista de emociones disponibles. Si se pulsa encima de una, en el cuadro blanco inferior aparece información relacionada con esta emoción, y si se mantiene pulsada se muestra como parte de la selección. Cuando se han seleccionado todas las emociones deseadas se puede dar al símbolo del disquete, que aparece abajo a la derecha, para guardar. Esto ocasiona que se muestre en pantalla un cuadro que permite escribir una breve descripción para el conjunto de emociones seleccionadas. Tras escribirla y seleccionar la opción de guardar se vuelve automáticamente a la pantalla principal, a la vez que aparece un mensaje que informa de que la etiquetación ha sido guardada.

El tercer elemento de la pantalla principal lleva a otra pantalla llamada practica. En esta pantalla al igual que en la anterior, la primera vez que se entra se muestra un mensaje informativo que se puede volver a visualizar de la misma manera que el otro. En esta pantalla se describe una situación y debajo se muestra la lista de emociones. Cada vez que

se selecciona una emoción aparece información sobre ella en el cuadro blanco inferior y si además formaba parte del conjunto de emociones predefinidas como solución de la situación, la emoción seleccionada se pone de color verde. Existe un botón, en la parte inferior central con el dibujo de una bombilla, que revela la solución de la situación, es decir, que todas las emociones que formaban parte de la solución se ponen de color verde. Existe otro botón en la parte inferior derecha con forma de flecha que sirve para pasar a la siguiente situación donde el funcionamiento será el mismo. Al cambiar de situación la lista de emociones se resetea, es decir, si había alguna seleccionada vuelve a su estado original. Una vez que se ha pasado por todas las situaciones que existen en la aplicación se llega a la pantalla principal de nuevo.

Por último en la pantalla principal se puede encontrar el enlace al historial. Si se entra sin haber guardado ninguna etiquetación aparece un mensaje que avisa de este hecho y si no, se muestra una lista con todas las etiquetaciones que se han guardado, indicando la descripción, la fecha, la hora y la lista de emociones de cada una de ellas.
Para terminar queda añadir que en todas las pantallas menos en la principal, la de elegir grupo y la del test, aparece un icono de una casa que tras pulsarlo lleva a la pantalla principal.